

SCR.2.2 TP 18 ⊥ :

Configuration DNS 2

bind9/Debian/IMUNES

<https://wiki.debian.org/Bind9>

Objectif. Mise en œuvre de serveurs primaire et secondaire sur une zone.

1. Se connecter comme **user1** et créer, directement dans le *domicile* de **user1**, le répertoire **IMUNES/TP18**
C'est dans ce répertoire que seront placés tous les fichiers créés pendant ce TP.
2. Si on n'est pas sur sa vm attitrée, alors à la fin de la séance, transférer le répertoire **TP18** vers son *domicile* personnel dans le répertoire **SCR.2.2/**
3. Lancer l'exécution de la topologie correspondant au fichier **dns2.imn** qui se trouve au même endroit que cette feuille d'énoncé (en plus du dessin, ici, en dernière page : il n'y a qu'à demander...).

I. Introduction

On va faire en sorte que les machines **host1**, ..., **host5**, **kheops**, **sethi**, soient des ressources d'un domaine TLD nommé **mos** sur lequel **sethi** est le dns primaire et **kheops** est un dns secondaire.

Dans le TP précédent, les clients DNS sur le LAN interrogent le serveur de noms indiqué dans leur fichier **/etc/resolv.conf**

Dans ce TP, les machines sont toutes censées être sur le domaine publique. Il n'y a pas beaucoup de sens à enregistrer l'adresse des dns de toutes les zones dans le fichier **/etc/resolv.conf** de chaque machine sur le net. On va alors faire en sorte que chaque machine s'adresse à un dns local à la machine même. C'est ce qui se passe lorsqu'on ne place rien dans le fichier **/etc/resolv.conf** (**man resolv.conf**). Ce sera donc le dns local à la machine qui recevra les requêtes avec récursion désirée provenant des applications qui tournent dans la machine. Ce dns local, recevant une telle requête, lancera alors le processus d'interrogation en commençant par s'adresser à un serveur de la racine de l'espace des noms de domaines.

Voici donc, en résumé, les étapes principales à réaliser pour arriver à cette architecture. Pour chacune d'elle, un paragraphe guide sera consacré.

1. Sur une machine témoin, disant **host10**, on indiquera :
 - (a) **ROOT-SERV** et son adresse dans le fichier qui contient les noms et adresses des serveurs de la racine de l'espace de nommage. Un tel fichier est fourni dans l'installation **bind** et peut être mis à jour par téléchargement (*<https://www.iana.org/domains/root/files>*)
On ne pourra pas l'utiliser tel quel puisque l'espace de nommage est fictif ici.
 - (b) dans le fichier de configuration de **named**, le nom du fichier précédent. C'est à partir de ce fichier que seront chargées dans le cache du dns local à la machine les informations sur les serveurs de la racine.
2. Sur **sethi**, on réalisera comme au TP précédent l'autorité sur **mos** directe et la zone inverse.
3. Sur **ROOT-SERV**, on créera le fichier de zone. C'est là qu'on fera référence à **mos**.

II. Sur ROOT-SERV, répertoire `/etc/bind/`

1. Créer le fichier zone racine en l'appelant `db.root.zone`
Indiquer que **ROOT-SERV** est SOA sur la zone racine, et que **sethi** et **kheops** sont serveurs de noms pour **mos.** et sa zone inverse.
2. Instruire à **named** qu'il est primaire sur la zone racine. Indiquer le nom `db.root.zone` là où il faut pour que **named** en ait connaissance.

III. Sur sethi, répertoire `/etc/bind/`

Comme on a fait au TP précédent,

1. Créer le fichier de zone directe `db.mos`, ainsi que le fichier `db.mos.inv` de zone inverse. Tester par **named-checkzone**.
2. Instruire à **named** qu'il est primaire sur les deux zones et lui indiquer le nom des fichiers de zones. Tester par **named-checkconf**.

IV. Sur host10, répertoire `/etc/bind/`

1. S'inspirer du contenu de `db.root` pour créer le fichier **hints-file** qui renseigne sur l'adresse de **ROOT-SERV** comme serveur de racine.
2. Indiquer à **named**, par l'intermédiaire du fichier `named.conf.default-zones`, le nom du fichier hints précédent.

V. Lancement et tests.

1. **named -g** lance le serveur de noms au premier-plan tout en envoyant tous les logs vers **stderr**. Sur quelle(s) machine(s) doit-on donner cette commande ?
2. **bind9** utilise DNSSEC : *Domain Name System Security Extensions* (<https://www.isc.org/dnssec/>). Cela permet au resolver recevant une réponse de s'assurer, à l'aide d'un mécanisme de vérification de signatures, que la réponse n'a pas été altérée pendant le transit. Un resolver attentif au DNSSEC (*DNSSEC-aware resolver*) doit être en possession d'une ancre de confiance (*trust anchor*) qui est, généralement, une clé publique pour la zone racine (<https://www.iana.org/dnssec/files>). On ne pourra pas utiliser cette ancre de confiance parce que l'espace de nommage ici est fictif. On peut créer sa propre clé publique pour la zone root fictive et l'indiquer aux resolvers. Pour ne pas compliquer encore ce TP, on va se contenter de désactiver l'option correspondante dans le fichier `named.conf.options` sur **host10**. **Avant de la désactiver :**
 - (a) Sur **host10**, passer la commande **dig host1.mos** puis la commande **dig +cd host1.mos**
 - (b) Consulter **man dig** pour comprendre le rôle de cette option de requête, et conclure.
3. **dig +trace** fait que le resolver correspondant à **dig** fait des requêtes itératives pour résoudre le nom, en suivant les références depuis les serveurs racine et en montrant la réponse de chaque serveur utilisé lors de la résolution.

```
root@host10:/# dig +trace host5.com
; <<>> DiG 9.10.3-P4-Debian <<>> +trace host5.mos
;; global options: +cmd
.                604614  IN      NS      ROOT-SERV.
;; Received 66 bytes from 127.0.0.1#53(127.0.0.1) in 0 ms

mos.             604800  IN      NS      sethi.mos.
```

```
;; Received 74 bytes from 90.90.0.10#53(ROOT-SERV) in 1 ms

host5.mos.          14400   IN      A       80.80.0.5
mos.                14400   IN      NS     sethi.mos.
;; Received 90 bytes from 80.80.0.20#53(sethi.mos) in 0 ms

root@host10:/# dig +trace -x 80.80.0.4
; <<>> DiG 9.10.3-P4-Debian <<>> +trace -x 80.80.0.4
;; global options: +cmd
.                  604442   IN      NS     ROOT-SERV.
;; Received 66 bytes from 127.0.0.1#53(127.0.0.1) in 0 ms

80.80.in-addr.arpa. 604800   IN      NS     sethi.mos.
;; Received 90 bytes from 90.90.0.10#53(ROOT-SERV) in 0 ms

4.0.80.80.in-addr.arpa. 14400   IN      PTR    host4.mos.
80.80.in-addr.arpa.  14400   IN      NS     sethi.mos.
;; Received 110 bytes from 80.80.0.20#53(sethi.mos) in 0 ms
```

VI. Sur kheops

1. Instruire à **named** qu'il est secondaire sur la zone **mos.** et la zone inverse, et lui indiquer le nom des fichiers où il gardera la copie des fichiers zones transférés depuis **sethi**. On gardera les mêmes noms : **db.mos** et **db.mos.inv** mais sans chemin absolu. Tester par **named-checkconf**.
2. Au début du fichier **named.conf.options**, un répertoire est indiqué. Consulter le contenu de ce répertoire. Maintenant lancer **named -g**, puis consulter de nouveau le contenu du répertoire précédent.
3. Demander par **dig** quels sont les serveurs de noms sur **mos.** Sur quel(s) nœud(s) cette commande est censée fonctionner ?
4. S'adresser directement à **kheops** par **dig** pour des requêtes sur la zone **mos.** Sur quel(s) nœud(s) cette commande est censée fonctionner ?
5. Sur **host10**, tenter par **dig** un transfert de zones depuis **sethi**. Le type correspondant d'une telle requête est AXFR. Cela fonctionne-t-il ? Si c'est le cas, alors il faudra y remédier parce que les transferts de zones surchargent le réseau. Il faut les laisser juste comme opérations de mises à jour à réaliser par les serveurs secondaires.
6. Incrémenter le numéro Serial pour simuler un changement dans la zone **mos**. Où fait-on cette incrémentation ?
7. Avant de relancer **named**, lancer une ligne de commande **tcpdump** pour voir les échanges qui vont avoir lieu entre **sethi** et **kheops**.

Si tout fonctionne correctement, alors copier les fichiers utilisés sur les nœuds virtuels vers **~user1/IMUNES/TP18/**, puis, si on n'est pas sur sa vm attitré, alors transférer par **sftp** le répertoire **~user1/IMUNES/TP18/** vers son compte personnel, dans le répertoire **SCR.2.2/** (**put -r ...**).

